VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Aplikace pro záznam a vizualizaci dat senzorů na mobilní platformě

Application for Sensor Data Recording and Visualisation on Mobile Platform

2019 Filip Šafránek

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Filip Šafránek**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Aplikace pro záznam a vizualizaci dat senzorů na mobilní platformě
Application for Sensor Data Recording and Visualisation on Mobile Platform

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je vytvořit aplikaci pro periodický záznam dat senzorů, poskytovaných platformou Android a dat z externího (externích) senzoru (senzorů). Platforma bude umožňovat vizualizaci uložených dat v mapě a doplňujících grafech, a jejich export do vhodného výměnného formátu, doplněného o dodatečné informace (např. TCX popř. GPX).

1. Prostudujte existující aplikace podobného zaměření a zjistěte, jaké možnosti poskytují a jak data zaznamenávají.
2. Nastudujte možnosti vizualizace měřených dat a možnosti jejich záznamu.
3. Vyberte vhodný externí senzor, zvolte implementační prostředí, popište knihovny, použité pro implementaci aplikace a jejich výběr zdůvodněte.
4. Analyzujte a navrhněte aplikaci.
5. Implementujte (multiplatformní) řešení pro záznam dat ze senzorů.
6. Aplikaci otestujte alespoň na dvou různých zařízeních, a pokud to bude možné, vhodným způsobem publikujte. Výsledky testů vyhodnoťte.

Seznam doporučené odborné literatury:

[1] Matt Gifford: PhoneGap Mobile Application Development Cookbook. Packt Publishing, 2013, 320 stran, ISBN: 978-1-84951-858-1.
[2] John M. Wargo: Apache Cordova 3 Programming. Addison-Wesley Professional, 2014, 262 stran, ISBN: 978-0-321-95736-8.
[3] Raymond Camden, Andy Matthews: jQuery Mobile Web Development Essentials. 2nd Edition, 2013, 242 stran, ISBN: 978-1-78216-789-1.
[4] Gabriel Svennerberg, Cameron Turner: Beginning Google Maps API 3. Apress, 2010, 328 stran, ISBN: 978-1-4302-2802-8.
[5] Meier, R.: Professional Android 4 Application Development, Wrox, 2012, ISBN-13: 978-1118102275
[6] SimpleLink™ SensorTag Software [online] [cit. 2016-02-15] Dostupné z <http://www.ti.com/tool/sensortag-sw?keyMatch=SensorTag&tisearch=Search-EN-TechDocs>

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Pavel Moravec, Ph.D.**

Datum zadání: 01.09.2018

Datum odevzdání: 30.04.2019

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 30. dubna 2019 .....

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

V Ostravě 30. dubna 2019 .....

Rád bych na tomto místě poděkoval Ing. Pavlu Moravcovi, Ph.D. za konzultace a odbornou pomoc při vytváření této bakalářské práce.

Abstrakt

Tato bakalářská práce se zabývá implementací nativní aplikace Stepcounter pro zpracování, záznam a export dat z vnitřních senzorů chytrého telefonu, a také pro záznam a grafické zobrazení dat z externího senzoru. Mobilní zařízení s touto aplikací je schopno sledovat fitness data uživatele jako třeba počet kroků nebo tyto data číst z externího senzoru. Dokáže taktéž sledovat jeho denní trasu, kterou je poté možno exportovat do formátu GPX. Práce také pojednává o základních informacích ohledně pohybových senzorů, mobilních platforem a multiplatformních aplikací, o technologii Bluetooth a systému GPS.

Klíčová slova: nativní aplikace, pohybové senzory, externí senzor, fitness, GPX, Bluetooth, bakalářská práce

Abstract

The aim of this bachelor thesis was to implement a native application Stepcounter for processing, recording and exporting data from the smart phone's internal sensors. and recording and graphically displaying data from an external sensor. Mobile device with this application is able to track user's fitness data or read this data from an external sensor and track its daily route, which it can then export to GPX. The thesis also discusses basic information about motion sensors, mobile platforms and cross platform applications, Bluetooth technology and GPS.

Key Words: native application, motion sensors, external sensor, fitness, GPX, Bluetooth, bachelor thesis

Obsah

Seznam použitých zkratek a symbolů

BT	–	Bluetooth
BLE	–	Bluetooth Low Energy
UUID	–	Universally Unique Identifier
HTML	–	Hypertext Markup Language
CSS	–	Cascading Style Sheets
JS	–	JavaScript
JSON	–	JavaScript Object Notation
XML	–	Extensible Markup Language
GPS	–	Global Positioning System
GPX	–	GPS Exchange Format
3G	–	třetí generace mobilních telekomunikačních technologií
4G	–	čtvrtá generace mobilních telekomunikačních technologií
GB	–	gigabyte
kbit	–	kilobit
IEEE	–	Institute of Electrical and Electronics Engineers
WLAN	–	Wireless Local Area Network
GATT	–	Generic Attribute Profile
ATT	–	Attribute Protocol
ID	–	Identifier
MIUI	–	MI User Interface
RAM	–	Random-access memory
USB	–	Universal Serial Bus
SD	–	Secure Digital
API	–	Application Programming Interface
SDK	–	Software Development Kit
W3C	–	World Wide Web Consortium
GUI	–	Graphical User Interface
WGS	–	World Geodetic System
PPS	–	Přesná polohová služba
SPS	–	Standardní polohová služba

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Obsahem této bakalářské práce je popis implementace nativní aplikace s názvem Stepcounter, která slouží pro zpracování, záznam a export dat z vnitřních senzorů chytrého telefonu, a také pro záznam a grafické zobrazení dat z externího zařízení Xiaomi Mi Band 2. Aplikace by měla číst data ze senzorů, ze kterých bude schopná počítat kroky, případně i rychlost a spálené kalorie. Bude také zaznamenávat denní polohu (trasu) pomocí GPS souřadnic na mapě. Tato data poté uchová po dobu jednoho týdne ve vnitřní paměti aplikace. Uložené pozice (trasy) bude schopná exportovat do formátu GPX. Aplikace bude číst počet kroků, informace o baterii a informace o aktivitách z externího BT zařízení Xiaomi Mi Band 2, které poté zobrazí v grafech nebo textově.

Motivací pro vytvoření aplikace Stepcounter bylo, že aplikací, které počítají kroky, je sice dostatečné množství, jenže během vývoje aplikace Stepcounter nebyla nalezena aplikace, která by zároveň také sledovala pozici na mapě a umožňovala export těchto dat do formátu GPX, byla schopná číst data z externího zařízení jako třeba Xiaomi Mi Band 2 a zároveň byla zcela zdarma. V dnešní době jsou fitness pomůcky, jako právě chytré náramky, velice populární. Tyto náramky vždy spolupracují s nějakou aplikací, pomocí které se čtou a zpracovávají data do čitelného formátu pro uživatele, je tedy zapotřebí vyvíjet aplikace tohoto typu.

V následující kapitole se budu věnovat popisu pohybových senzorů v mobilních zařízeních, popisu externího senzoru Xiaomi Mi Band 2 a technologiím pro komunikaci s tímto senzorem. Dále se podívám na aktuální mobilní platformy a přiblížím, co to jsou multiplatformní aplikace a také možnosti jejich vývoje. Další kapitola se poté bude věnovat GPS systému, jazyku XML a formátu GPX, který se systémem GPS souvisí. Potom proberu některé z podobných aplikací aplikaci Stepcounter. Poté se už budu věnovat analýze a návrhu aplikace, následované samotnou implementací. Nakonec proberu testování a jak celkově dopadl vývoj této aplikace.

2 Senzory a komunikace

V této kapitole jsou popsány pohybové senzory, které se nacházejí v mobilních zařízeních a další senzory nacházející se v externím zařízení Xiaomi Mi Band 2. Dále se v této kapitole budu věnovat samotnému zařízení Xiaomi Mi Band 2 a technologiím pro komunikaci s ním.

2.1 Senzory

Jsou zařízení, které detekují nebo měří fyzikální vlastnosti a posílají tyto informace dál, většinou do nějakého procesoru.

2.1.1 Pohybové senzory

V mobilních zařízeních je pro snímání pohybu možno využít tří různých senzorů: akcelerometr, gyroskop a magnetometr. Aby bylo možné detekovat pohyb co možná nejpřesněji, využívá se většinou spojení těchto senzorů. Informace o těchto senzorech podle Ulip Marek[12] a mobilmania.cz[1].

Akcelerometr – Je senzor, který sleduje zrychlení telefonu, při změně rychlosti, ať už z nulové či nenulové, zaznamená tuto změnu.
Funkčnost akcelerometru podle literatury[12]: „*Při změně dochází k vibracím, spojených (sic!) tímto pohybem. Akcelerometr využívá mikroskopické krystaly, na kterých se při působení vibrací tvoří napětí, odpovídající danému zrychlení. Tomu se říká piezoelektrický jev, díky kterému je možné určit směr gravitace a tím i natočení přístroje.*“

Gyroskop – Poskytuje data o natočení telefonu vzhledem k výchozí pozici, nebude-li se zařízeni pohybovat, nevrací žádné hodnoty. Spojením gyroskopu s akcelerometrem lze dosáhnou přesnějšího určení směru pohybu.

Magnetometr – Je měřicí přístroj, který měří magnetické pole ve třech osách. Bývá tvořen zmagnetizovaným kouskem plechu, u kterého dochází ke změně odporu vlivem magnetického pole. Změna odporu se vyhodnocuje elektrickým obvodem. Využívá se jako kompas pro určení světových stran. Magnetický senzor je využíván i GPS navigací, kde se jeho použití projeví rychlejším určením polohy. V kombinací s akcelerometrem a gyroskopem tak lze získat ještě přesnější určení směru pohybu.

Existují také senzory jako odometr nebo tachometr, ty však nejsou běžně vestavěny v mobilních zařízeních.

Odometr – Je přístroj používaný především ve vozidlech sloužící k měření uražené vzdálenosti. Může být elektronický, mechanický nebo kombinací těchto dvou typů.

Tachometr – Je zařízení sloužící k měření rychlosti, tento přístroj se podobně jako odometr používá většinou ve vozidlech.

2.1.2 Xiaomi Mi Band 2

Xiaomi Mi Band 2 je chytrý náramek od firmy Xiaomi (obr. 1). Náramek umožňuje měření kroků, srdečního tepu a sledování aktivit, jako třeba průběh spánku, chůze, běh, atd.

Pohybový senzor – Zařízení obsahuje již zmíněné pohybové senzory viz kapitola 2.1.1, pomocí nichž je schopen počítat kroky, či zjistit jakou aktivitu zrovna uživatel vykonává (chůze, běh, atd.).

Optický snímač tepové frekvence – Je zařízení pro snímání srdeční tepové frekvence. Snímač používá fotopletyzmografií. Tato metoda využívá změny světelné propustnosti tkáně při změně tlaku krve. Je to založeno na skutečnosti, že světlo bude rozptýleno předvídatelným způsobem. Tato změna je pak závislá na tepové frekvenci srdce. [2]

Obrázek 1: Xiaomi Mi Band 2 [8]

2.2 Komunikace se senzorem

2.2.1 Technologie

Pro komunikaci mezi mobilním a externím zařízením se dnes většinou využívají bezdrátové technologie. V mobilních zařízeních se používají 3G a 4G sítě, WiFi technologie nebo Bluetooth.

Infračervená bezdrátová komunikace je dnes v mobilních zařízeních už trochu zastaralá, a tak jí nové přístroje většinou ani nepodporují, proto jí nebudu ani popisovat.

3G síť – Neboli třetí generace mobilních telekomunikačních technologií. Standard 3G nabízí možnost bezdrátového přenosu hlasu a dat s rychlostí alespoň 200 kbit/s.

4G síť – Čtvrtá generace mobilních telekomunikačních technologií. Standard 4G nabízí zvýšenou rychlost přenosu dat s rychlostí alespoň 100 Mbit/s.

WiFi – Označení pro několik standardů IEEE 802.11 popisujících bezdrátovou komunikaci v počítačových sítích. Technologie využívá tak zvaného „bezlicenčního frekvenčního pásma". Zařízení kompatibilní s technologií Wi-Fi se mohou připojit k Internetu prostřednictvím sítě WLAN a bezdrátového přístupového bodu.

Bluetooth – Je otevřený standard pro bezdrátovou komunikaci propojující dvě a více elektronických zařízení, jako například mobilní telefon a chytrý náramek.

Pro komunikaci s chytrým náramkem se používá technologie Bluetooth, konkrétně Bluetooth Low Energy. Hlavním důvodem, proč se používá tato technologie, je vysílací výkon, který je mnohonásobně nižší než u ostatních technologii, pohybuje se v řádu mW. Z nízkého vysílacího výkonu plyne nízká spotřeba elektřiny.

Od verze 4.0 byla tato technologie rozdělená na tři části, Bluetooth Classic, Bluetooth High Speed a Bluetooth Low Energy.

- **Classic Bluetooth** – Skládá se ze starších Bluetooth protokolů.
- **Bluetooth High Speed** – Kombinuje Bluetooth a WiFi, aby dosáhl vysokých přenosových rychlostí, vyžaduje čip kombinující Bluetooth a WiFi, který je poměrně drahý.
- **Bluetooth Low Energy (BLE)** – Nazývaná také Bluetooth Smart je inteligentní, výkonná verze bezdrátové technologie Bluetooth. Dělá chytrá zařízení kompaktní, cenově dostupná a méně složitá.

  Tato technologie byla spuštěna jako součást specifikace Bluetooth 4.0 Core. Původním cílem bylo poskytnout rádiový standard s nejnižší možnou spotřebou energie, specificky optimalizovanou pro nízké náklady, spotřebu, složitost a nízkou šířku pásma,.

  Problémy u Bluetooth Classic jsou rychlé vybíjení baterie a častá ztráta spojení, což vyžaduje časté párování. Úspěšné řešení těchto problémů je jedním z důvodů rychlého přijetí BLE. Dalším důvodem je obrovské rozšíření chytrých telefonů a tabletů.

  Technologie zaměřená hlavně na aplikace ve zdravotnickém, fitness, bezpečnostním a zábavním průmyslu. [3]

2.2.2 Komunikace přes BLE

Při popisu této komunikace jsem se inspiroval články od autorů Saurabh Pant a Shahar Avigezer, a také Android dokumentací [4, 5, 6]. Velice důležité je, že během komunikace může být BLE zařízení připojeno pouze k jednomu centrálnímu přístroji.
Klíčovými komponentami, jak jsou definovány Android Developers[6], jsou:

- **Generic Attribute Profile (GATT)** – Je obecná specifikace pro odesílání a přijímání krátkých dat, známých jako atributy, přes BLE spojení. Všechny současné profily Low Energy aplikací jsou založeny na GATT. Definuje způsob výměny dat pomocí předem definovaných atributů.

- **Attribute Protocol (ATT)** – Na vrcholu atributového protokolu (ATT) je postaven GATT. Toto je také označováno jako GATT/ATT. ATT je optimalizován pro běh na zařízeních BLE. Každý atribut je jednoznačně identifikován identifikátorem UUID (Universally Unique Identifier), což je standardizovaný 128bitový formát pro ID řetězce, který se používá k jednoznačné identifikaci informací. Atributy transportované ATT jsou formátovány jako vlastnosti a služby.

- **Charakteristika** – Obsahuje jednu hodnotu a 0-n deskriptorů, které popisují hodnotu charakteristiky. Charakteristiku lze považovat za typ, který je analogický s třídou. Charakteristiky mají vždy nastavená oprávnění jako jsou read, write nebo notify.

- **Služba** – Je sbírka charakteristik. Například byste mohli mít službu s názvem „Heart Rate Monitor“, která zahrnuje charakteristiku jako „heart rate measurement“.

- **Deskriptory** – Jsou definované atributy, které popisují charakteristiky. Například deskriptor může specifikovat popis čitelný pro člověka, přijatelný rozsah pro hodnotu charakteristiky nebo měrnou jednotku, která je specifická pro hodnotu charakteristiky. Je to volitelný atribut, ne každá charakteristika je jim popsaná.

Zařízení BLE může mít jeden nebo více různých profilů GATT pro více účelů, jako je detekce aktivit, počítání kroků nebo měření tepové frekvence.

Pro komunikaci s BLE přístrojem je třeba znát UUID služby a charakteristiky, ze které mohou být data získána, protože zařízení neposílá data samostatně, ale je třeba o ně požádat.

3 Použité technologie a vývoj v mobilním prostředí

3.1 Mobilní platformy

Mobilní platforma je speciální operační systém určený především pro mobilní zařízení, jako jsou chytré telefony, tablety, atp. Systém je většinou nahrán na zvláštní interní, a pro běžného uživatele nepřepisovatelné, paměti. Rozdíl mezi platformami je v poskytovaném operačním systému uživateli, rozhraním operačního systému a také řadou aplikací třetích stran, které jsou na dané platformě k dispozici. Dnes jsou nejvíce používané platformy Google Android a Apple iOS.

3.1.1 Android

Android je mobilní operační systém založený na jádře Linuxu, který je dostupný jako otevřený software. Jeho vývoj vede společnost Google. Používá se na chytrých telefonech, tabletech a dalších zařízeních. Název je často doplněn o název prostředí, které vyvíjí výrobce zařízení (například MIUI od Xiaomi). Nejnovější verze androidu je aktuálně 9.0 s kódovým označením Pie, která vyšla 6. srpna 2018. Vyšla také verze 10 s kódovým označením Q, ale prozatím pouze v Beta verzi a jenom pro zařízení Pixel. Nejvíce rozšířené verze jsou v rozmezí od 4.4 až 8.1, jak jde vidět v tabulce 1. Údaje v této tabulce byly nashromážděny během 7-denního období končícího 26. října 2018.

Během implementace jsem pracoval s platformou Android, proto teď aspoň ve zkratce popíšu, co přinesly jednotlivé verze od 4.4 do 9.0. Informace o verzích Androidu podle pcmag.com[9] a Forrest Stroud[10].

- **4.4 KitKat** – Úhlednější rozhraní a další změny, včetně vylepšení služby pro okamžité zasílání zpráv, úpravu fotografií a režim zobrazení na celé obrazovce. Podpora starších telefonů s méně než 1 GB paměti RAM. Bylo přidáno „OK Google“.
- **5.0 Lollipop** – Přepracované uživatelské rozhraní, známé jako Material Design, zlepšena kontinuita v zařízeních se systémem Android, podpora více uživatelů, možnost uživatelského účtu hosta a nový systém oznámení. Běhový modul Java Dalvik byl nahrazen systémem Android Runtime.
- **6.0 Marshmallow** – Přidán mobilní platební systém Android Pay a standardizovaná podpora otisků prstů, automatické zálohování dat do Google Cloudu, podpora USB typu C, lepší správa aplikací, větší povědomí o kontextu uživatele v aplikaci Google Now a integrace karty SD do interního úložiště.
- **7.0 Nougat** – Pokud je aplikace kompatibilní, lze ji měnit a pohybovat na obrazovce jako u stolního počítače. Nabídka „Nastavení“ ukáže více informací bez přechodu na druhou úroveň, informace o nouzových situacích se můžou zobrazit na obrazovce uzamčení. Režim

Tabulka 1: Zastoupení verzí Androidu v mobilních zařízeních [7]

Verze	Kódové označení	API	Zastoupení na trhu
2.3.3 - 2.3.7	Gingerbread	10	0.2%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.3%
4.1.x	Jelly Bean	16	1.1%
4.2.x	Jelly Bean	17	1.5%
4.3	Jelly Bean	18	0.4%
4.4	KitKat	19	7.6%
5.0	Lollipop	21	3.5%
5.1	Lollipop	22	14.4%
6.0	Marshmallow	23	21.3%
7.0	Nougat	24	18.1%
7.1	Nougat	25	10.1%
8.0	Oreo	26	14.0%
8.1	Oreo	27	7.5%

virtuální reality podporuje nadcházející funkci VR od společnosti Google a režim Doze zlepšuje výdrž baterie.

- **8.0 Oreo** – Zlepšuje životnost a výkon baterie tím, že omezuje aplikacím práci na pozadí. Uživatelé mají větší kontrolu nad upozorněním a automatickým vyplňováním. Přináší podporu Bluetooth 5 a funkci Wi-Fi Aware.

- **9.0 Pie** – Zlepšuje životnost baterie tím, že nabízí adaptivní funkce, jako je naučit se používat aplikace a měnit jas. Pie monitoruje využití času obrazovky a také předpovídá, co uživatel udělá dál. Lze připojit až pět zařízení Bluetooth.

3.1.2 iPhone OS (iOS)

Operační systém iOS je mobilní systém vytvořený společností Apple. Tento systém je použit výhradně pro zařízení Apple (iPhone, iPad, atd.). V současnosti je nejnovější verzí iOS 12. iOS je odlehčenou verzí operačního systému Mac OS X, mobilní verze má navíc podporu dotykového ovládání. Pro iOS se vyvíjí v programovacím jazyku Objective-C, což je určitá nadstavba C. iOS je poměrně uzavřený systém, který má mnoho omezení a přísná pravidla při vývoji, obecně se má za to, že je to bezpečnější systém než Android.

3.2 Multiplatformní aplikace

3.2.1 Definice

Podle literatury [11] je multiplatformní software „*takový počítačový software, který je implementován pro několik různých počítačových platforem, jakými mohou být Windows, Linux apod. Multiplatformní software může být rozdělen do dvou typů.*

První vyžaduje individuální build nebo kompilaci pro každou platformu, která je podporována. Druhým typem je software, který může být přímo spuštěn na kterékoliv platformě bez nutnosti buildu nebo kompilace. Také jinak interpretovaný jazyk nebo předkompilovaný přenosný bytecode pro které jsou interpreti nebo běhové prostředí společné nebo standardní součásti na všech platformách.“

3.2.2 Multiplatformní vývoj

Jedná se o vývoj softwarových produktů pro více platforem nebo prostředí. Pro tento druh vývoje lze použít různé metody k přizpůsobení softwaru různým operačním systémům. [11]

Existuje několik nástrojů, které umožňují vývoj mobilních aplikací, ty úspěšné a použitelné se dají rozdělit do tří základních skupin.

1. **Nativní aplikace** – Pro vývoj se používá SDK (Software Development Kit) určený přímo pro danou platformu. Java a Android SDK na Androidu, Swift či Objective-C pro iOS. [13]

2. **Nativní multiplatformní aplikace** – Je napsána nástrojem, který umožňuje její překlad do nativních jazyků a umožňuje tak její spuštění na více platformách se sdíleným zdrojovým kódem. Nejpoužívanějším nástrojem je Xamarin. Ten umožňuje vyvíjet v jazyce C# nejen pro mobilní platformy, ale umožňuje také sdílet výsledný kód mezi operačními systémy pro stolní počítače (Windows, macOS). [13]

3. **Hybridní aplikace** – Je kombinací webových a nativních technologií. Vychází z principu interpretace HTML kódu, který je ale zapouzdřen v nativní aplikaci. Jedná se tedy o webovou stránku, která se zobrazuje uvnitř samostatné aplikace. Nepoužívá nativní ovládací prvky dané platformy, ale HTML prvky, které se na jednotlivých platformách liší. Nejpoužívanějším nástrojem je Apache Cordova. [13]

Při vypracovávání této práce jsem vytvářel hybridní aplikaci, proto se dále zaměřím pouze na tuto skupinu.

3.2.2.1 Vývoj hybridních aplikací

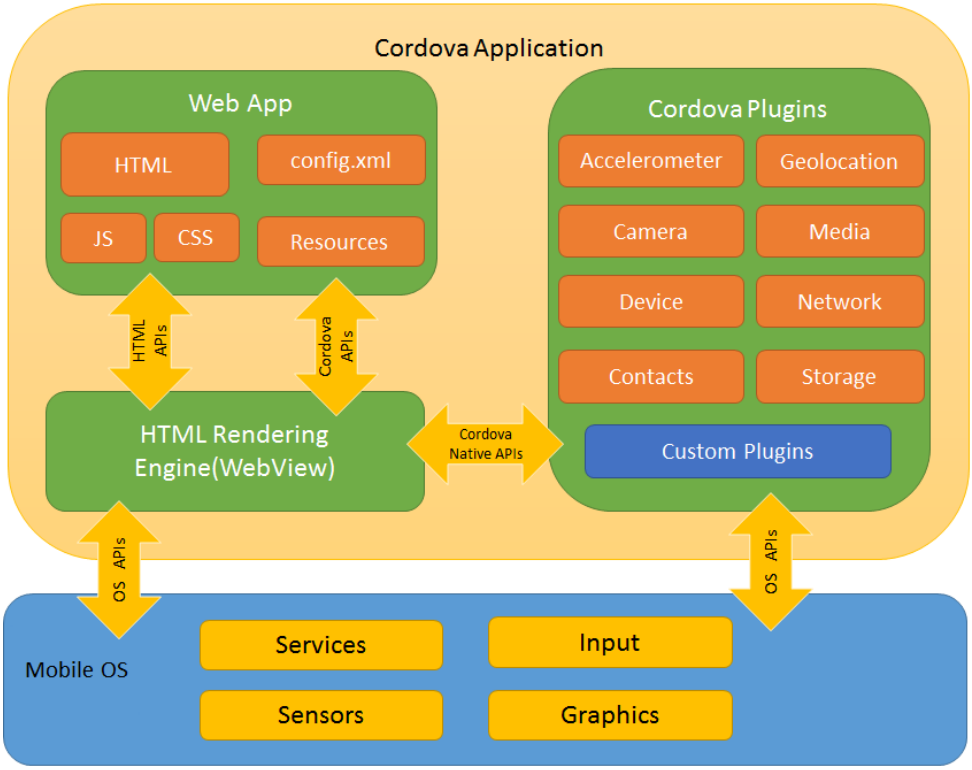
HTML, CSS a JS tvoří třídu základních technologií pro World Wide Web.

HTML (Hypertext Markup Language) – Je to standardní jazyk pro tvorbu webových stránek a webových aplikací. Definuje význam a strukturu webového obsahu. „Hypertext“ označuje odkazy, které propojují webové stránky, a to buď v rámci jedné, nebo mezi různými webovými stránkami. Jazyk je charakterizován množinou značek a jejich vlastnostmi. Těmito značkami jsou vymezené prvky HTML a určují tak význam obsahu stránky. Prvky HTML jsou nejzákladnějšími stavebními kameny webových stránek.[15]

CSS (Kaskádové styly) – Je jazyk používaný k popisu vzhledu dokumentu napsaného značkovacím jazykem jako HTML nebo XML. Je navržen tak, aby umožnil oddělení prezentace a obsahu, včetně rozvržení, barev a písem. CSS je standardizován napříč webovými prohlížeči podle specifikace W3C. Název kaskádové pochází ze zadaného schématu priority, pokud tedy jeden prvek splňuje více pravidel, je dáno, které pravidlo se použije. [16]

JS (JavaScript) – Je skriptovací jazyk, který se používá především na webu. Je to skriptovací jazyk na straně klienta, což znamená, že zdrojový kód se většinou zpracovává ve webovém prohlížeči klienta, než na webovém serveru. JavaScript je interpretovaný jazyk, nemusí být tedy kompilován. Webové stránky jsou vykreslovány dynamicky, díky tomu mohou stránky reagovat na události, vykazovat speciální efekty, přijímat proměnné texty, atd. JavaScript je jazyk používaný k poskytování interakce se stránkou, nebo jednoduše k zavedení funkčnosti na stránku. Jelikož JavaScript pracuje většinou s HTML stránkami, vývojář potřebuje znát HTML, aby mohl tento skriptovací jazyk plně využít. [14]

Apache Cordova – Je open-source framework pro mobilní vývoj. Umožňuje vývoj multiplatformních aplikací použitím webových technologií jakými jsou HTML, CSS a JavaScript. Aplikace jsou spouštěny v rámci balíčků zacílených na jednotlivé platformy. Přes API, které jsou kompatibilní se standardy, mají přístup ke schopnostem mobilních zařízení, jako jsou data ze senzorů, data v zařízení, stav sítě atd. Cordova podporuje tyto platformy: Android, iOS, OS X a Windows. Cordova aplikace má několik složek. Obrázek 2 ukazuje pohled na architekturu Cordova aplikace. [17]



Obrázek 2: Architektura Cordova aplikace [17]

3.2.2.2 API a Pluginy (plug-in)

API – Je sada rutin, protokolů a nástrojů pro vytváření softwarových aplikací. Obecně se jedná o soubor definovaných metod komunikace mezi různými komponentami. Používají se při programování komponent grafického uživatelského rozhraní tzv. GUI. Dobré API usnadňuje vývoj programu tím, že poskytuje všechny stavební bloky, které poté programátor sestavuje. Účelem je zjednodušení programování tím, že API odstraní základní implementaci a vystaví objekty, které vývojář může využít.[18]

Plugin – Softwarová komponenta, která přidává specifický prvek do existujícího počítačového programu. Pokud program podporuje pluginy, umožňuje přizpůsobení. Jednoduše to je doplněk programu, který k němu přidává funkce. Jsou běžně používány v internetových prohlížečích, ale mohou být využity také v jiných typech aplikací. [19]

Pro vytvoření této práce jsem využil pluginy pro Cordovu, a to: Background Mode, Bluetooth LE, File, Geolocation, Network Information, Core Motion Pedometer, Screen Orientation a Whitelist, které dále popíšu. K popisu jednotlivých pluginů byly použity jejich dokumentace. [20]

- **BackgroundMode** – Plugin pro provádění aplikace na pozadí. Většina aplikací nemusí běžet na pozadí, proto v režimu na pozadí pozastaví aplikaci a obnoví ji až před přepnutím

do režimu popředí. Systém poté uchovává všechna síťová připojení otevřená v pozadí, ale nedoručuje data, dokud se aplikace neobnoví.

Na většině mobilních operačních systémů není podporován nekonečný běh aplikace na pozadí, a proto aplikace s touto funkčností nemusí být přijaty veřejnými obchody jako třeba Google Play.

- **Bluetooth LE** – Plugin umožňuje komunikaci se zařízeními Bluetooth Low Energy.
- **File** – Tento plugin implementuje File API umožňující přístup k souborům umístěným v zařízení. Je založen na několika specifikacích, včetně: HTML5 File API, také implementuje FileWriter specifikaci.
- **Geolocation** – Tento plugin poskytuje informace o poloze zařízení. Zdroje informací o poloze, ze kterých plugin čerpá, zahrnují systém GPS a umístění odvozené ze síťových signálů. Neexistuje však žádná záruka, že API poskytne aktuální polohu zařízení.
- **Network Information** – Poskytuje informace o připojení zařízení (WiFi, mobilní připojení nebo zda vůbec má připojení k internetu).
- **Core Motion Pedometer** – Načítá údaje z vestavěného krokoměru v zařízení, jako je počet kroků.
- **Screen Orientation** – Cordova plugin pro nastavení orientace obrazovky běžným způsobem pro iOS, Android a Windows. Tento plugin je založen na API pro orientaci obrazovky.
- **Whitelist** – Tento plugin implementuje zásady whitelist pro navigaci webového prohlížeče aplikace Cordova 4.0 a vyšší. Je určen pouze pro Android, protože zdrojový kód pluginu sám podporuje pouze platformu Android.

  Whitelisting domény je model zabezpečení, který řídí přístup k externím doménám, nad nimiž aplikace nemá žádnou kontrolu.

3.3 GPS a GPX

3.3.1 GPS

Systém GPS, neboli Globální polohový systém, je satelitní radionavigační systém pro určování času na Zemi. Systém je ve vlastnictví Spojených států amerických (USA). GPS poskytuje 24 hodin denně kdekoliv na Zemi signály, které GPS přijímače dokážou zpracovat a určit z nich polohu a přesný čas. Systém GPS se skládá ze tří základních segmentů: kosmického, řídícího a uživatelského. [21]

1. **Kosmický segment** – V současné době tvořen 28 satelity na šesti oběžných drahách. Družice obíhají ve výšce cca 20 200 km s inklinací 55 stupňů a doba oběhu je přibližně

12 hodin. Z jakéhokoli místa na světě by v daný okamžik měly být viditelné alespoň 4 družice. Může jich být až 12. Pro určení polohy je potřeba přijímat signály z minimálně 4 družic, jakákoliv další družice zlepšuje přesnost. [21]

2. **Řídící segment** – Tvořen monitorovacími stanicemi po celém světě. Monitorovací stanice sbírají data a následně je předávají do hlavní řídící stanice. Zde se data zpracovávají, vypočtou se přesné údaje o oběžných dráhách a provede se korekce času, zpracované data se potom pošlou zpátky do satelitů. Satelity pak data vysílají a ty jsou přijímány GPS přijímači. [21]

3. **Uživatelský systém** – Je tvořen GPS přístroji, které poskytují údaje o poloze, rychlosti a čase uživatelům na Zemi pomocí různých aplikací. [21]

GPS využívá základního geocentrického souřadného systému WGS-84 (World Geodetic System). Systém je z roku 1984 a poskytuje údaje o zeměpisné délce a šířce. Systém GPS poskytuje dvě úrovně služeb. První je PPS (Přesná polohová služba), poskytuje přesná data, ale pouze autorizovaným uživatelům, jakými jsou armáda USA nebo NATO. Druhou úrovní je SPS (Standardní polohová služba), dostupná všem po celém světě.

3.3.2 XML a GPX

XML – Značkovací jazyk a je také W3C doporučenou specifikací jako univerzální značkovací jazyk. Na rozdíl od jiných značkovacích jazyků XML není předdefinován, takže při použiti se definují vlastní značky. U jazyka je kladen důraz na jednoduchost, obecnost a hlavně použitelnost napříč různými systémy. Hraje důležitou roli při výměně dat na webu i jinde. Na základě XML je vytvořeno mnoho formátů a je také možné vytvořit si svůj vlastní. Jazyk popisuje především strukturu dokumentu a nezabývá se vzhledem. Vzhled může být definován pomocí CSS. [23, 24]

GPX – Datový formát XML pro výměnu dat GPS mezi aplikacemi a webovými službami. Podporovanými verzemi jsou aktuálně GPX 1.0 a 1.1. Jelikož se jedná o XML formát, data o poloze jsou uložena ve značkách. Formát je otevřený a může být použit bez nutnosti platit licenční poplatky. [22]

4 Podobné aplikace

Na trhu je aktuálně řada fitness aplikací, představím zde ty nejoblíbenější (nejpoužívanější). Loga aplikací byla převzata z jejich oficiálních stránek v obchodě Google Play.

4.1 Google Fit

Aplikace počítá kroky, minuty v pohybu a sleduje cíle, které si uživatel nastaví. Podporuje také Wear OS od společnosti Google. Má možnost synchronizace s oficiální aplikaci od Xiaomi, takže je schopna číst informace ze Xiaomi náramků. Aplikaci jde propojit s jinými aplikacemi tohoto druhu. Během hledání informací o této aplikaci jsem se dozvěděl, že počítání kroků je hodně nepřesné, což jsem si i sám ověřil a většina jiných dostupných aplikaci je proto lidmi preferovanější, vyplývá to i z hodnocení v obchodě Google Play kde má aplikace 3.8/5 hvězdiček (ke dni 22/3/2019).

4.2 Leap Fitness Step Counter and Pedometer

Aplikace je spíše jednodušší provedení krokoměru, která nevyžaduje žádnou registraci, službu Cloud a ani sledování GPS. Obsahuje volitelnou zálohu na Google Disk a přehled informací v grafech. U aplikace existuje profesionální verze, která odstraňuje reklamy, za kterou je však třeba zaplatit. V obchodě Google Play má velmi slušné hodnocení 4.7/5 hvězdiček (ke dni 22/3/2019).

4.3 MyFitnessPal

Je jednou z výkonnějších aplikací krokoměru, umí však více věcí jako počítání kalorii, fitness sledování, hlídání stravy (třeba sacharidy) a pitný režim. Aplikace je částečně zdarma, ale pro rozsáhlejší databázi a lepší funkce je třeba opět zaplatit. Někteří uživatelé mají s aplikací různé technické problémy (u nejnovější verze), i přesto v obchodě má pořád hodnocení 4.6/5 hvězd (ke dni 22/3/2019).

4.4 Pedometer

Je další aplikace z jednodušších aplikací pro měření kroků. Aplikace je vhodná třeba pro plánované tréninky, obsahuje tlačítko start/stop, má tedy možnost sledování pohybu, když uživatel chce a ne neustále. Sleduje také spálené kalorie, vzdálenost, rychlost a rozeznává aktivity chůze/běh. V obchodě má hodnocení 4.4/5 hvězdiček (ke dni 22/3/2019).

4.5 Pacer Health Pedometer and Step Counter

Jedna z populárních aplikací pro počítání kroků. Umožňuje ladění citlivosti krokoměru hned několika způsoby v „nastavení krokoměru“. Aplikace obsahuje GPS sledování polohy, podporu pro Fitbit i MyFitnessPal. Obsahuje premium funkce například pro tzv. koučing. V obchodě má hodnocení 4.6/5 hvězdiček (ke dni 22/3/2019).

4.6 Runkeeper

Hlavním cílem aplikace je v podstatě sledování uživatelových procházek a běhů. Nabízí sledování pohybu, cíle, fitness rutiny, zprávy o pokroku. Zahrnuje také různé funkce pro motivační účely jako výzvy komunity. Aplikace také má svou prémiovou verzi, která by měla zajišťovat lepší sledování průběhu a podporu živého sledování (live tracking). Hodnocení v obchodě je 4.5/5 hvězd (ke dni 22/3/2019).

4.7 Runtastic Steps

Je další z populárních aplikací pro počítání kroků. Bezplatná verze zahrnuje sledování kroků, integraci se službou Google Fit, přehledy aktivit a podporu dalších aplikací a přenosných zařízení, sleduje také spálené kalorie. V placené verzi dokáže sledovat spánek, cyklistiku a obecné fitness funkce. V obchodě má hodnocení 4.3/5 hvězdiček (ke dni 22/3/2019).

5 Analýza a návrh

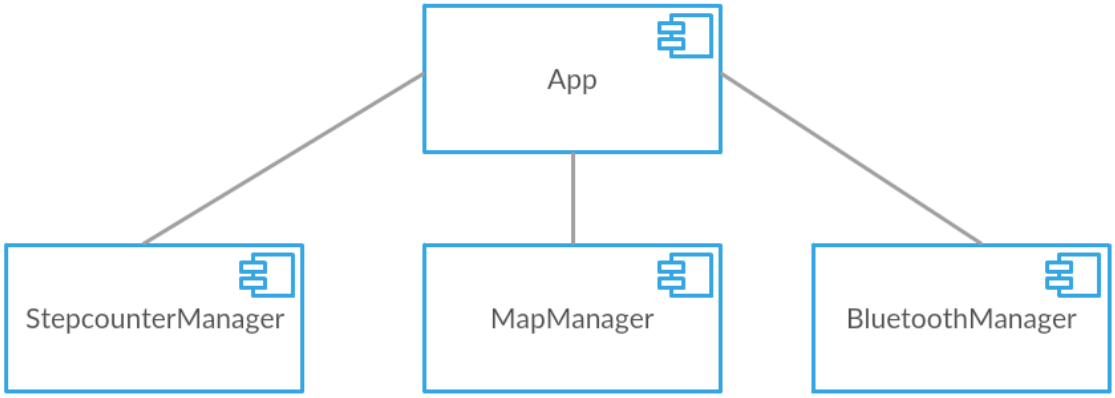
5.1 Výběr verze Androidu

Technologie Bluetooth LE je podporována až od verze Androidu 4.3. Podle tabulky 1 má však tato verze aktuálně zastoupení v mobilních zařízeních pouze 0.4%, proto stačí aby aplikace Stepcounter podporovala až verze Androidu 4.4 a vyšší.

5.2 Návrh aplikace

Od zapnutí aplikace neustále poběží služby pro počítání kroků a sledování pozice. Další funkčnosti, jimiž jsou generování GPX souboru a BT komunikace, bude spouštět uživatel.

App je hlavní části aplikace, která zprostředkovává vizualizaci a propojení dat, StepcounterManager se stará o počítání kroků a všechny operace s tím spojené, MapManager zajišťuje sledování pozice zařízení a práci s mapou a BluetoothManager se stará o komunikaci s externím senzorem přes BLE.



Obrázek 3: Diagram komponent

5.3 Výběr vývojového prostředí

Pro vývoj aplikace jsem si zvolil cestu hybridních aplikací, konkrétně vývoj přes Apache Cordova. Hlavní výhodou je to, že stačí napsat jeden kód, který poběží na všech platformách, stejně jako webové stránky. Není tedy nutné vytvářet více různých kódů pro různé platformy, jak tomu je u nativního vývoje.

Apache Cordova je jedním z nejpoužívanějších a nejznámějších prostředí tohoto druhu, dá se tedy předpokládat, že má rozsáhlou komunitu vývojářů, a proto jsem zvolil právě Cordovu.

U výběru verze Cordovy záleží na tom jaké verze jsou potřeba pro jednotlivé pluginy. Nejnovější verze Core Motion Pedometer pluginu vyžaduje verzi alespoň 5.0.0, stejně jako plugin pro

BLE komunikaci nebo pro geolokaci. Avšak pluginy běží stejně i na novějších verzích, proto je třeba jenom hlídat, aby verze byla vyšší než 5.0.0. Pro vývoj aplikace byla použita verze 9.0.0, protože je aktuálně nejnovější a měla by tak podporovat všechny pluginy.

5.4 Návrh uživatelského rozhraní

5.4.1 Hlavní nabídka (Main Menu)

Slouží k pohybu uživatele v aplikaci, nabídka obsahuje nejenom prvky pro pohyb v aplikaci, ale také řídicí tlačítka, která se můžou na různých stránkách lišit. Tlačítka sloužící k pohybu v aplikaci jsou 4 a vyskytují se na každé stránce, jedná se o:

1. **Home** – toto tlačítko odkazuje na hlavní stránku a zajišťuje tak přesun uživatele na hlavní stránku aplikace.
2. **Map** – tlačítko uživatele přepne na zobrazení stránky s mapou.
3. **Devices** – zobrazí stránku pro vyhledávání bluetooth zařízení.
4. **Connected Device** – zobrazí stránku s již připojeným (zapamatovaným) zařízením.

Na každé stránce se také vyskytuje tlačítko **Refresh** pro obnovení stránky. Na hlavní stránce se ještě objeví tlačítko **Generate GPX**. Toto tlačítko slouží k vytvoření GPX souboru v interní paměti zařízení (telefonu) na cestě /storage/emulated/0/Android/data/com.stepcounter.app/files. Tento soubor obsahuje data o poloze z posledních 7 dní, pokud aplikace tyto data obsahuje, jinak se vygeneruje bez dat. Aplikace ukládá vždy jen jeden tento soubor, to znamená, že starý soubor, přepíše aby uživatel třeba neopatrností zbytečně nezabíral paměť zařízení stejnými soubory.

5.4.2 Hlavní strana (Home)

Na hlavní straně bude zobrazeno počítání kroků a další informace vyplývající ze získaných (spočtených) dat jako množství spálených kalorii, uražená vzdálenost nebo průměrná rychlost během probíhajícího dne.

Dále na stránce bude tabulka posledních 7 dnů, kde po zvolení konkretního dne, se nám zobrazí počet kroků v tomto dni. Den se v tabulce zobrazí, pouze pokud má aplikace data z daného dne, tedy například pokud máme aplikaci teprve 2 dny (a aplikace byla spuštěna a mohla tak zaznamenat data), bude zde tabulka pouze s těmito 2 dny.

Dole bude také zobrazen ukazatel průběhu denního cíle, nastaven na 5000 kroků.

5.4.3 Mapa (Map)

Pro vytvoření mapy je zapotřebí využít nějaké externí knihovny nebo API. Vybíral jsem mezi Google Maps API a OpenLayers knihovnou. Na mapu nebyly žádné zvláštní požadavky, prakticky pouze zaznamenávání pozice na mapě, takže i přesto, že Google Maps má více možností,

stejně by se nevyužily, OpenLayers je jednodušší a nepotřebuje žádnou registraci, navíc obrovskou výhodou je, že je zcela zdarma, na rozdíl od Google Maps, které po překročení daného limitu využití mapy vyžadují placení za službu. OpenLayers mají také dobře napsanou dokumentaci a spoustu příkladů, jak lze s mapou pracovat. Byla tedy vybrána knihovna OpenLayers.

Stránka pro mapu obsahuje jednoduše pouze mapu. Na mapě bude vyobrazena trasa z probíhajícího dne. Mapa je interaktivní, to znamená, že uživatel s ní může pohybovat, nebo jí třeba přibližovat a oddalovat.

5.4.4 Vyhledávání zařízení (Devices)

Stránka pro vyhledávání BT zařízení v okolí. Po přepnutí na tuto stránku se automaticky spustí vyhledávání (scanning). Pokud uživatel nemá zapnutý Bluetooth, aplikace to zahlásí, a vyhledávání vůbec nezačne. Pro vyhledání zařízení tedy uživatel bude muset Bluetooth zapnout a stránku obnovit.

Aplikace vyhledá pouze zařízení, ke kterým je schopna se připojit a číst z nich data, může to tedy chvíli trvat, a zobrazí je v tabulce. Vyhledávání pokračuje, dokud uživatel neopustí stránku, nebo nevybere zařízení, ke kterému se chce připojit. Po vybrání zařízení se objeví zpráva o uložení adresy a od té chvíle bude možno získat data na stránce připojeného zařízení. Pokud aplikace vyhledává zařízení příliš dlouho, třeba už pár minut a pořád bezvýsledně, je třeba buďto obnovit stránku nebo i restartovat aplikaci. Čas vyhledávání zařízení se odvíjí od toho, kolik BT zařízení je v okolí.

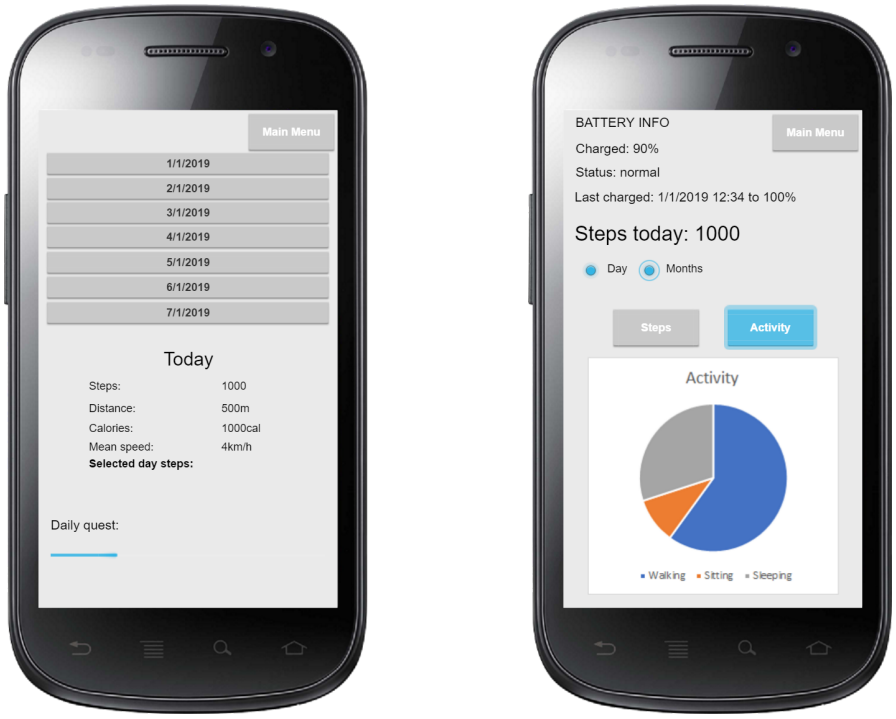
5.4.5 Připojené zařízení (Connected Device)

Na teto stránce se zobrazí data z připojeného BT zařízení. Pro získání dat je opět potřeba aby byl Bluetooth zapnutý. Pokud je BT vypnutý, aplikace nenatáhne žádná nová data, ale pokud již byla nějaká data zaznamenána dříve, zobrazí se grafy s těmito daty. Když je naopak zapnutý, začnou se stahovat aktuální data a zobrazí se nahoře informační nápis o stahování dat. Až nápis zmizí, znamená to, že jsou data stažená a pro jejich zobrazení v grafech je potřeba stránku obnovit.

Budou zde také zobrazené informace o stavu baterie zařízení, jako na kolik je nabitá, v jakém se nachází stavu (normální nebo jestli se zrovna nabíjí), a datum posledního nabíjení s informací, na kolik procent byla nabitá. Stejně tak zde bude i informace o aktuálně uražených krocích.

Grafy budou rozdělené do skupiny s daty z posledních 24 hodin (Day) a s ostatními daty z minimálně posledních 30 dní (Months), pokud nějaká taková data jsou. Tyto kategorie jsou vyobrazeny pomocí tzv. radio buttonu, který lze přepínat a vybrat tak zobrazovanou kategorii.

Aplikace bude vytvářet 2 druhy grafů, a to spojnicový a koláčový graf. Spojnicový pro zobrazení dat o krocích (Steps), kde bude vyobrazen vždy počet kroků za hodinu. Koláčový graf zase bude použit u druhů aktivit, kde je procentuálně vyjádřeno, jak často uživatel v daném čase konkrétní aktivity prováděl.



(a) Návrh hlavní stránky

(b) Návrh stránky pro data z BT zařízení

Obrázek 4: Návrh vzhledu hlavní stránky a stránky pro data z BT zařízení

5.5 Volba vytváření grafů

Jak již bylo zmíněno aplikace bude vytvářet 2 druhy grafů, spojnicový pro počet kroků a koláčový pro aktivity. Pro vytvoření grafů je zapotřebí použít externí knihovny, protože JavaScript ze základu nemá žádné funkce pro vytváření grafů. U spojnicového grafu je potřeba, aby graf byl schopen pracovat s velkým množstvím bodů a aby byl interaktivní. Uživatel by tedy mohl zobrazit počet kroků ve zvoleném bodě nebo přiblížit (zvětšit) konkrétní oblast. Koláčový graf nemusí pracovat se spoustou dat, ale měl by být přehledný a interaktivní.

Knihovna D3.js nemá ze základu přidanou interaktivitu a Chart.js má horší práci s větším objemem dat. Byla proto vybrána knihovna Highcharts, protože dokáže pracovat s velkým množstvím dat a interaktivitu podporuje ze základu, není tedy třeba vytvářet žádné funkce navíc pro tuto funkčnost.

6 Implementace

Vytváření hybridní aplikace je vlastně to samé jako vytváření webové stránky, jak již bylo zmíněno v kapitole 3.2.2.1. Podstatou jsou tedy HTML, JS a CSS soubory. V HTML dokumentu je uvedena struktura aplikace, v JS dokumentu zase funkčnost a CSS udává vzhled aplikace. Nejobsáhlejším a také vlastně nejdůležitějším je JS dokument, protože jednoduše bez něj by aplikace nic nedělala. Samozřejmě je možné zavést přímo JS kód do HTML dokumentu ve značce „<script>“, ale bylo by to velice neestetické a nepraktické, jelikož by se v kódu nevyznal ani samotný vývojář.

Budu se věnovat postupně každému dokumentu a funkčnosti v něm, začnu funkčností aplikace, kterou umožňuje JS dokument, poté se budu věnovat struktuře aplikace, čili HTML a nakonec se krátce podívám na vizuální styl aplikace, který udává CSS.

6.1 Funkčnost (JS)

6.1.1 Práce na pozadí

Jedním z hlavních úkolů aplikace je počítání kroků a tato funkčnost musí běžet, i když uživatel aplikaci zrovna nepoužívá, čili aplikace je zapnutá, ale běží na pozadí. Funkčnost zajišťuje plugin Background mode pro Cordovu, který stačí na začátku běhu aplikace aktivovat (povolit) a on se postará, aby se aplikace na pozadí neuspala.

```
cordova.plugins.backgroundMode.enable();
cordova.plugins.backgroundMode.on('activate', function() {
   cordova.plugins.backgroundMode.disableWebViewOptimizations();
});
```

Výpis 1: Práce na pozadí

6.1.2 Krokoměr

V této kapitole se budu věnovat hlavní funkčnosti aplikace, a to čtení dat z vnitřních senzorů zařízení a následné zpracování těchto informací. Ne na každém zařízení je možné počítaní kroků, pro počítání kroků musí zařízení obsahovat senzory pro snímání pohybu viz kapitola 2.1.1.

V aplikaci jsem použil dvě metody počítání kroků. První metodou je získávání dat z vnitřního krokoměru (pokud ho zařízení má) a druhou metodou je získávání dat pomocí vestavěné funkčnosti přes DeviceMotionEvent, která jednoduše vrací data z pohybových senzorů, které je třeba dále zpracovat.

Metoda s daty z vestavěného krokoměru

Pro získání dat z vestavěného krokoměru jsem použil plugin Core Motion Pedometer Plugin for

Cordova. Přidání pluginu do projektu se provádí pomocí příkazu v příkazové řádce „cordova plugin add cordova-plugin-pedometer“. Slova „cordova plugin“ specifikují, že se bude jednat o plugin pro Cordovu, „add“ říká, že plugin budeme k projektu přidávat. Je samozřejmě možnost plugin odstranit, či vypsat seznam pluginů již obsažených v projektu a „cordova-plugin-pedometer“ je jednoduše název pluginu, který se má přidat. Po spuštění příkazu se plugin nejdříve stáhne a poté nainstaluje do projektu. Tento plugin vytváří globální objekt „pedometer“, tento objekt obsahuje funkce, pomocí kterých se lze dostat k datům, které potřebujeme. Většina pluginů pro Cordovu funguje právě takhle, takže u dalších již nebudu rozepisovat jejich zavedení do projektu.

Jak již bylo zmíněno, na začátku se ověřuje, zda zařízení umí počítat kroky. To se provádí pomocí jedné z metod objektu „pedometer“.

```
pedometer.isStepCountingAvailable(successCheckStepcounter,
    errorCheckStepcounter);
```

Výpis 2: Ověření zda zařízení umí počítat kroky

Pokud zařízení umí počítat kroky, ověřuje se, zda má možnost číst data z vestavěného krokoměru (jestli má vestavěný krokoměr). To se prověří pouze tím, jestli předchozí metoda vrátila nějaká data. Pokud ano, pak jsou to data z krokoměru, pokud nevrátí nic, znamená to, že zařízení nemá vestavěný krokoměr.

Po tom, co je ověřeno, že zařízení má potřebné senzory a také krokoměr, je možno začít číst tyto data a začít je zobrazovat. Zavolá se metoda, která sleduje krokoměr a když ten zachytí nová data, metoda nám tyto data poskytne ke zpracování. Metoda nevrací každý jednotlivý krok, ale vždy skupinu kroků v čase. V aplikaci jsem tedy vytvořil místo v lokálním úložišti (localStorage), kde se ukládají kroky z posledních 7 dnů, jedná se o pole objektů, kde každý objekt představuje právě jeden den. Pro uložení kroků tedy stačí vždy vzít toto pole z lokálního úložiště, najít v poli objekt pro aktuální den (nebo ho vytvořit pokud ještě neexistuje) a přidat právě přijatá data.

V lokálním úložišti se dá ukládat data pouze ve formátu textového řetězce, proto je třeba před uložením pole objektů serializovat do JSON formátu a při čtení zase textový řetězec ve formátu JSON konvertovat zpět na pole objektů. Toho se dá docílit pomocí funkcí JSON.stringify a JSON.parse.

```
pedometer.startPedometerUpdates(successHandler, onError);

var successHandler = function (pedometerData) {
   let retrievedObject = localStorage.getItem('stepsData');
   let objH = JSON.parse(retrievedObject);
   for(var i = 0; i < objH.length; i++){
```

```
                if(objH[i].day == today){
                obj = objH[i].data;}
            }
        numberOfSteps = obj.step + numberOfSteps;
        dObject = {
                            step:numberOfSteps,
                            stepSize:podo_stepSize,
                            weight:podo_weight,
                            height:podo_height};
        saveStepcounterDataLocalStorage(dObject);
    }
```

Výpis 3: Čtení dat z krokoměru

Pro zobrazení dat na displeji je třeba prvně získat HTML element, do kterého budeme data zapisovat a potom je tam jednoduše přiřadíme.

Metoda s daty pouze ze senzorů

U této metody jsem použil kód s algoritmem pro výpočet kroků od Sébastien Ménigot. [25] Detekce kroků je založena na změně X, Y, Z souřadnic z DeviceMotionEventu. Pokud tedy aplikace detekuje „houpavý“ pohyb charakteristický pro chůzi či třeba běh, zaznamená to jako krok. Zároveň aplikace měří tyto kroky v čase, takže je schopná spočítat rychlost a spálené kalorie. Kód však reaguje na jakýkoli houpavý pohyb ve všech třech osách je tedy velice jednoduché aplikaci ošálit třeba točením zařízení v ruce (což je také detekováno jako chůze).

Pro zlepšení robustnosti je použit Kalmanův filtr, ale jeho funkčnost není nijak zvlášť znát. Ve zjednodušené podobě jde o predikční-estimační algoritmus, který se pokouší předpovědět na základě předchozích vzorků signálu vzorky následující. Předpovídaná data jsou poté porovnána s naměřenými daty a použity u další předpovědi. [26]

6.1.3 Mapa

Jak již bylo zmíněno v analýze pro vytvoření mapy, byla zvolena knihovna OpenLayers.

Z oficiálních stránek OpenLayers se dá stáhnout knihovnu s mapou a začít ji používat v kódu. Prvním krokem je samozřejmě vytvoření mapy. V aplikaci jsem tedy vytvořil třídu Map, která přes konstruktor vytvoří mapu a pomocí své funkce Add dokáže přidávat čáry na tuto mapu.

Konstruktor požaduje 3 argumenty: cíl, pozici a přiblížení. Cíl představuje id elementu v HTML, ve kterém bude mapa vytvořena (většinou nějaký div). Pozice je kde má být mapa na začátku centrována, kde se má načíst, protože mapa se samozřejmě nenačítá celá, ale pouze oblast, kterou

chceme vidět. Přiblížení neboli zoom poté představuje, jakou oblast uvidíme (jaká se načte), je to jednoduše přiblížení mapy.

```
this.Map = new ol.Map({
        layers: [this.Raster, this.Vector],
        target: this.Target,
        view: new ol.View({
      center: this.Position,
      zoom: this.Zoom,
      projection: 'EPSG:4326'})});
```

Výpis 4: Vytvoření mapy

Funkcí Add se volá s 1 argumentem, a to čárou, kterou chceme nakreslit na mapu. Čáru jednoduše představují 2 body na mapě, které se na mapě spojí a vznikne tak čára.

```
Add(data){
   this.Source.addFeature(new ol.Feature( new ol.geom.LineString(data) ));}
```

Výpis 5: Kreslení čár do mapy

6.1.4 Grafy

Grafy slouží k vizualizaci dat a obzvláště velkého množství dat, které aplikace Stepcounter čte z externího senzoru. Jak již bylo zmíněno v kapitole 5.5, pro vytváření grafů byla použita knihovna Highcharts. Po naimportování potřebných Highcharts knihoven, můžeme začít grafy vytvářet. Pro vytvoření spojnicového grafu jsem napsal funkci createGraph, která přijímá 4 argumenty: data, divID, title a yAxisName.

- **data** – jsou jednoduše data, která mají být zanesená do grafu.
- **divID** – udává element (div) v HTML, do kterého bude graf vytvořen a také zobrazen.
- **title** – tímto argumentem nastavíme nadpis, který bude u grafu, neboli nějaký popisek grafu.
- **yAxisName** – určuje název (popisek) osy Y.

Na začátku funkce se nastaví velikost elementu, do kterého vytváříme graf, protože nechceme, aby se graf roztáhl libovolně přes celou stránku. Ideální je nastavování dle velikosti okna nebo dokumentu, aby aplikace zůstala responzivní.

```
var h = $(window).height();
var w = $(window).width();

var div = document.getElementById(divID);
div.style.height = (h/2)-30 +'px';
div.style.width = w-30 +'px';
```

Výpis 6: Nastavení velikosti místa pro graf

Graf samotný se vytváří pomocí metody Highcharts.chart z Highcharts knihovny. Nastavení typu grafu, nadpisu, popisu os a dalších věcí se posílá jako objekt do argumentu této metody. U spojnicového grafu osa X je časová osa, Highcharts má pro časovou osu připravené nastavení, které vyžaduje připojení další knihovny s časovými zónami. Poté už stačí jen nastavit časovou zónu, která bude na ose X zobrazena.

```
Highcharts.setOptions({
   time: {timezone: 'Europe/Prague'}});
```

Výpis 7: Highcharts nastavení časové zóny

Spojnicový graf je základním typem grafu pro Highcharts, proto není třeba nastavovat typ grafu. Na osu X stačí nastavit, že typ dat bude datetype a Highcharts tam nastaví časovou zónu podle nastavení, které bylo přidáno výše.

```
{chart: {zoomType: 'x'},
     title: {text: title},
     xAxis: {type: 'datetime'},
     yAxis: {
           title: {text: yAxisName}},
series: [{
           type: 'area',
           name: 'steps in time',
           data: data}]}
```

Výpis 8: Ukázka objektu poslaného do argumentu funkce pro vytvoření grafu

Vytváření koláčového grafu je obdobné jenom tady už je třeba zvolit typ grafu.

```
chart: {type: 'pie'}
```

Výpis 9: Nastavení typu grafu

Spojnicový graf očekává data udávající body, které budou poté zobrazeny (vykresleny) v grafu. V aplikaci tedy posílám řadu bodů, která je v kódu reprezentována polem polí (dvourozměrné

pole), kde vnitřní pole mají velikost 2. Každý bod se vlastně skládá ze souřadnice X a Y, kde X představuje čas a souřadnici Y představují data v daném čase.

V případě aplikace Stepcounter je spojnicový graf použit u znázornění počtu kroků v čase. Kroky jsou zaznamenány na grafu vždy co hodinu, takže funkce sčítá kroky do hodinových intervalů a pak tyto součty i s danou hodinou ukládá do pole, které poté pošle do funkce pro vytvoření grafu.

```
//Steps (per hour)
if(obj.minute[c].time < (helpTime + hour)){
   stepsPerHour += obj.minute[c].steps;}
else{
   allDataSteps.push([helpHour,stepsPerHour]);
   helpTime = obj.minute[c].time;
   d = new Date(helpTime);
   d = new Date(d.getFullYear(),d.getMonth(),d.getDate(),d.getHours());
   helpHour = d.getTime();
   stepsPerHour = obj.minute[c].steps;}
```

Výpis 10: Výpočet kroků za hodinu

Koláčový graf čeká na data, odkud získá název skupiny a počet výskytů (celé číslo). V aplikaci je to řešeno opět polem polí, kde vnitřní pole mají velikost 2. U koláčového grafu však není tolik dat, konkrétně v aplikaci Stepcounter je těchto polí 6 a to **walking** pro aktivitu chůze, **running** pro aktivitu běhu, **charging** pro nabíjení baterie, **sleeping** pro spánek, **standing** pro aktivitu kde uživatel pouze stál a **sitting** pro sezení. Druhou hodnotou je poté množství výskytu dané aktivity v čase.

```
allDataActivity.push(["WALKING",walkingCounter]);
allDataActivity.push(["RUNNING",runningCounter]);
allDataActivity.push(["CHARGING",chargingCounter]);
allDataActivity.push(["SLEEPING",sleepingCounter]);
allDataActivity.push(["STANDING",standingCounter]);
allDataActivity.push(["SITTING",sittingCounter]);
```

Výpis 11: Množství výskytu jednotlivých aktivit

Získané data se poté pošlou do funkcí pro vytvoření grafů.

```
createGraph(allDataSteps,'chart','Steps in time','Steps');
createPieChart(allDataActivity,'chart2');
```

Výpis 12: Vytvoření grafů

6.1.5 Generování GPX souboru

Soubor GPX se generuje z uložených dat o pozicích a uloženého počtu kroků z jednotlivých dnů. Pro uložení dat o krocích, rychlosti a směru pohybu, je potřeba v GPX využít element „extensions“, který umožňuje přidat vlastní, ještě nedefinovaná data. Pro podporu tohoto elementu je potřeba využít verze 1.1. Vytváření GPX dokumentu je vytváření XML, takže se definují elementy, kterým se přiřazují atributy nebo hodnoty.

```
var doc = document.implementation.createDocument("", "", null);
var gpx = doc.createElement("gpx");
gpx.setAttribute("version","1.1");
gpx.setAttribute("creator","safranek vsb");
var a = JSON.parse(localStorage.getItem('mapCoords'));
if(a){
   var trk = doc.createElement("trk");
   var stepsEx = doc.createElement("extensions");
   var stepsE = doc.createElement("steps");
   stepsE.innerHTML = stepsCount;
   stepsEx.appendChild(stepsE);
   trk.appendChild(stepsEx);
   for(var j = 0; j < a[i].data.length; j++){
      var trkpt1 = doc.createElement("trkpt");
      trkpt1.setAttribute("lat",a[i].data[j][1]);
      trkpt1.setAttribute("lon",a[i].data[j][0]);
      trk.appendChild(trkpt1);}
   gpx.appendChild(trk);}
doc.appendChild(gpx);
docWrite = new XMLSerializer().serializeToString(doc);
```

Výpis 13: Generování GPX

6.1.6 Práce s interním úložištěm

Pro ukládání a čtení souboru z interního úložiště zařízení jsem použil plugin cordova-plugin-file zmíněný v kapitole 3.2.2.2. Plugin definuje globální objekt „cordova.file“, pomocí kterého lze později pracovat s úložištěm. Aplikace Stepcounter používá interní úložiště pro ukládání dat z externího senzoru, k těmto datům uživatel za běžných okolností (nemá práva roota v mobilu) nemá přístup. Interní úložiště je také použito pro ukládání vygenerovaného GPX souboru, ke kterému však už uživatel samozřejmě přístup musí mít, takže se data uloží na jiné místo v úložišti, kde uživatel má práva přístupu.

Prvním krokem je vybrání místa kam se soubor uloží nebo odkud budeme číst. To se provádí v prvním argumentu funkce „resolveLocalFileSystemURL“. Úložiště lze určit pomocí objektu poskytnutým File pluginem. Dále se vybere soubor podle jména, přes funkci getFile, pokud soubor neexistuje tak se vytvoří. Když je soubor nalezen můžeme s ním pracovat.

```
window.resolveLocalFileSystemURL(cordova.file.externalDataDirectory,
    function (dirEntry) {
   var isAppend = false;
    //create file
   dirEntry.getFile(fileName, {create: true, exclusive: false}, function(
       fileEntry) {
      //write file
      writeFileCordova(fileEntry, docWrite, isAppend);
   }, errorCallback);
}, errorCallback);
```

Výpis 14: Práce s úložištěm

Pro zapisování do souboru se používá FileWriter objekt, u kterého zavoláme metodu write.

```
fileEntry.createWriter(function (fileWriter) {
   fileWriter.onwriteend = function() {
      console.log("Successful file write...");};
   fileWriter.onerror = function (e) {
      console.log("Failed file write: " + e.toString());};
   fileWriter.write(dataObj);});
```

Výpis 15: Zapisování do souboru

6.1.7 Bluetooth

Jak již bylo zmíněno v kapitole 2.2, s náramkem se komunikuje pomocí technologie BLE. Pro tuto komunikaci byl použit plugin BluetoothLE. Tento plugin podporuje spoustu funkcí, které lze dělat s BLE zařízeními jako vyhledávání, připojování a popisování zařízení nebo čtení a zapisování dat. Většinou při čtení a zapisování dat nečteme (nezapisujeme) data v surové podobě, ale data jsou nějakým způsobem kódována, tyto funkce plugin také podporuje.

Postup komunikace:

1. **Inicializace**
   Jde o zjištění zda vůbec zařízení, na kterém běží aplikace obsahuje BT. Je to logický první

krok zahájení komunikace, protože pokud zařízení nemá BT nemá smysl pokračovat v jakýchkoli dalších krocích. Pro inicializaci je použita metoda „initialize“, ke které máme přístup přes objekt „bluetoothle“ z výše zmíněného pluginu. Většina metod, které obsahuje tento objekt požaduje jako argumenty funkce, které se spustí při úspěchu nebo při neúspěchu, často je také možnost poslat jako třetí argument nějaké nastavení (tzv. options). Metoda také zjistí zda je BT zapnutý nebo vypnutý, jelikož opět pokud je BT vypnutý nemá smysl pokračovat v následujících krocích.

```
var initializeSuccessCallback = function(data){
   if(data.status == "disabled"){
      alert("Bluetooth is off");
      return;}}
bluetoothle.initialize(initializeSuccessCallback,
    initializeErrorCallback, {request: true});
```

Výpis 16: Inicializace BT

2. **Skenování (vyhledávání)**
   Jedná se o vyhledávání všech BT zařízení v okolí, ke kterým se bude možno pokusit připojit. Skenování je provedeno pomocí metody „startScan“, tato metoda jednoduše vyhledává všechny BT zařízení v okolí a vrací základní informace o nich, jako třeba adresu a jméno. Nejpodstatnějším údajem je samozřejmě adresa zařízení, protože pomocí ní se pak k zařízení lze připojit a pracovat s ním.

   Aplikace v této metodě vytváří seznam BT zařízení z okolí a poté je prozkoumává a uživateli nakonec zobrazí v tabulce pouze ty, které jsou pro aplikaci použitelné. Ukládá se seznam zařízení, aby se jedno zařízení neprozkoumávalo zbytečně vícekrát.

```
var devices = [];
var startScanSuccessCallback = function(data){
   var BTdevice = {address: data.address,
      name: data.name,}
   var repeat = false;
   devices.forEach(function(element) {
      if(element.address == data.address){
         repeat = true;}});
   if(!repeat)
      devices.push(BTdevice);}
bluetoothle.startScan(startScanSuccessCallback, startScanErrorCallback)
```

Výpis 17: Vyhledávání BT zařízení

Pro prozkoumání BT zařízení je třeba se k němu nejdříve připojit a pro připojení je zase potřeba, aby žádné BT zařízení již nebylo připojeno, takže pokud je, musíme ho odpojit. Vhodné zařízení je v aplikaci určeno podle toho, zda při prozkoumání obsahovalo charakteristiky pro čtení kroků nebo aktivity, které jsou dále v aplikaci využity.

```
bluetoothle.discover(function(data){
   data.services.forEach(function(el){
      for (let e of el.characteristics) {
         if(e.uuid.toLowerCase() == "00000005-0000-3512-2118-0009
             af100700" || e.uuid.toLowerCase() == "
             00000007-0000-3512-2118-0009af100700"){
            var column = $('<tr>');
            column.append($('<td>').append('<button type="button" value=
                '+BTdevice.address+'>'+BTdevice.name+" "+BTdevice.address
                +'</button>'));
            table.append(column);}}
   });}
```

Výpis 18: Vytváření tabulky vhodných BT zařízení

3. **Připojování**
   Jednoduše připojení se k BT zařízení. Pro připojení jsem použil metodu „connect“ z pluginu, která zajišťuje čistě připojování a nahlášení statusu například, že se zařízení připojilo.

```
bluetoothle.connect(connectSuccessCallback, connectErrorCallback, {
    address: connectedAddress});
```

Výpis 19: Připojení BT zařízení

4. **Prozkoumání (discover)**
   Když už jsme k zařízení připojeni, lze jej prozkoumat. To znamená podívat se, jaké služby, charakteristiky a deskriptory zařízení poskytuje. Lze tak zjistit, co vlastně zařízení umí. Tato metoda je zobrazena ve výpisu kódu č.18.

5. **Práce se zařízením**
   Tady se už věnujeme přímo práci se zařízením, jako je třeba čtení či zápis dat. Metoda z pluginu pro čtení se jmenuje „read“ a pro zápis „write“. Metodu pro zápis jsem v aplikaci používal jako žádost o nějaké data. Pokud o data žádám, je třeba mít na charakteristice, od které chceme data, povolený odběr dat (tzv. subscribe). To zajišťuje funkce „subscribe“ z pluginu.

Metoda pro čtení je v aplikaci využita pro čtení dat o baterii a aktuálním počtu kroků. Informace o baterii můžeme číst vždy, není třeba, aby naše zařízení bylo autentizováno u BT zařízení, proto tato funkce funguje vždy, když jsme k němu připojení. Aktuální počet kroku je však braný jako důležitější údaj a pro jeho získání je potřeba, aby naše zařízení již bylo autentizováno. Autentizace se provádí zasláním požadavku na autentizační charakteristiku zařízení, u Mi Band 2 to je charakteristika 00000009-0000-3512-2118-0009af100700 ve službě FEE1. U této charakteristiky musí být povolený také odběr dat.

Autentizace u zařízení Mi Band 2 probíhá tak, že do charakteristiky pro autentizaci pošleme data délky 18 bajtů, kde první dva bajty mají hodnoty 1 a 8, a dalších 16 bajtů tvoří autentizační klíč. Po odeslání těchto dat čekáme na odpověď, která, pokud proběhne vše v pořádku, bude začínat třemi bajty s hodnotami 16, 1 a 1, po obdržení této odpovědi pošleme do zařízení data velikosti dvou bajtů s hodnotami 2 a 8, čímž žádáme o náhodný autentizační klíč. Zase čekáme na odpověď zařízení, která by měla začínat třemi bajty s hodnotami 16, 2, 1. Následující data obsahují autentizační klíč, který je však AES kódovaný musíme ho tedy dekódovat (proto je v aplikaci importována knihovna pro AES kódování). Do zařízení pošleme zpět dekódovaný klíč a před něj vložíme dva bajty s hodnotami 3 a 8 pro dokončení autentizace. Zařízení by poté mělo vrátit odpověď začínající třemi bajty s hodnotami 16, 3, 1 a potvrdit tak, že autentizace proběhla úspěšně. Pokud nastane při autentizaci chyba, zařízení vrací data začínající bajty 16, 3, 4.

Metody pro čtení, zápis i povolení odběru musí mít zadané, odkud mají číst nebo kam zapisovat. Tyto údaje se zadávají do třetího argumentu jako objekt. Data která jsou zapotřebí jsou adresa, služba a charakteristika. Data, které posíláme do zařízení nebo naopak čteme je potřeba zakódovat/dekódovat.

```
var params = {
   address: connectedAddr,
   service: "FEE0",
   characteristic: "00000007-0000-3512-2118-0009af100700" //REALTIME
      STEPS}
bluetoothle.read(function(data){
   var batD = bluetoothle.encodedStringToBytes(data.value);
   let numOfSteps = batD[1];
   let numOfSteps1 = 256 * batD[2];
   let numOfSteps2 = 65536 * batD[3];
   let ns = numOfSteps + numOfSteps1 + numOfSteps2;
   var text = document.createTextNode("Steps today: " + ns);
}, readError, params);
```

Výpis 20: Čtení dat z BT zařízení

Data, které nám BT zařízení pošle na požádání, se vrací do metody „subscribe“ (jak již bylo řečeno, musíme mít povolený odběr dat pomocí této metody). To, že se jedná o zachycené data, poznáme v metodě podle statusu, který má hodnotu „subscribedResult“. Mi Band 2 ukládá data pro jednotlivé minuty, jedná příchozí hodnota (u charakteristiky pro ACTIVITY DATA) představuje 4 krát 4 minuty, takže v ní přijdou 4 hodnoty, kde každá představuje 4 minuty dat ze zařízení. Na začátku každé této série je identifikační číslo těchto dat, které při čtení můžeme ignorovat.

```
var subscribeSuccessActivity = function(data){
   if(data.status == "subscribed"){
      console.log("subscribed activity data");}
   else if(data.status == "subscribedResult"){
      var actData = bluetoothle.encodedStringToBytes(data.value);
      var activity, steps;
      var minData = new Uint8Array(4);
      for(j = 0; j < 4; j++){
         for(i = 0; i < 4; i++){
            minData[i] = actData[(i+1)+(j*4)];}
         timeSent += 60000; //1 minute
         activity = minData[0];
         steps = minData[2];
         dataArray.push({time:timeSent,activity:activity,steps:steps})
             ;}}}
```

Výpis 21: Zachycené data metodou subscribe

Metoda pro zápis dat funguje podobně jako předchozí metody s tím rozdílem, že je třeba do ní poslat data, která zapisujeme. Tyto data se zadávají opět do třetího argumentu metody do atributu objektu „value“.

```
let par = {
   address: connectedAddr,
   service: "FEE0",
   characteristic: "00000004-0000-3512-2118-0009af100700", //[
      WriteWithoutResponse", "Notify"]} UUID_CHAR_FETCH
   value: encodedString,
   type: "noResponse"}
bluetoothle.write(writeSuccess, writeError, par);
```

Výpis 22: Posílání dat do BT zařízení

6. **Odpojení**
   Pokud již dále s BT zařízením nechceme pracovat, tak jej odpojíme pomocí metody „disconnect“.

```
bluetoothle.disconnect(disconnectSuccessCallback,
    disconnectErrorCallback, {address: er.address});
```

Výpis 23: Odpojení BT zařízení

7. **Uzavření spojení**
   Jedná se vlastně o pokračování odpojování BT zařízení. Je to definitivní ukončení spojení, hlavně u starších verzí pluginu bylo potřeba prvně odhlásit zařízení a poté zrušit spojení, u nových verzí již stačí pouze odpojení zařízení nebo rovnou zrušení spojení. Provádí se metodou „close“.

```
bluetoothle.close(closeSuccessCallback, closeErrorCallback, {address:
    er.address});
```

Výpis 24: Uzavření spojení

6.2 Implementace základního GUI (HTML)

HTML je vlastně to, co vidí uživatel, proto je potřeba do něj importovat všechny potřebné skripty a CSS soubory, které bude aplikace neboli náš soubor HTML využívat.

6.2.1 Import dokumentů

Pro funkčnost aplikace Stepcounter je zapotřebí importovat hned několik externích knihoven. Knihovny pro možnost použití jQuery metod a příkazů, pro práci s mapou a samotné vytvoření mapy, pro práci s Bluetooth, knihovny umožnující vytváření grafů, také knihovny pro AES šifrování, vytváření notifikačních oken, detekci a počítání kroků a v neposlední řadě také Cordova a vlastní soubory.

```
<script src="js/jquery-2.1.3.min.js"></script>
<link rel="stylesheet" type="text/css" href="jquery.mobile-1.4.5/jquery.
   mobile-1.4.5.css">
<link rel="stylesheet" type="text/css" href="openLayers/ol.css">
<script src="openLayers/ol.js"></script>
<script src="js/highcharts.js"></script>
<script src="js/moment.min.js"></script>
<script src="js/moment-timezone-with-data-2012-2022.min.js"></script>
<link rel="stylesheet" type="text/css" href="css/index.css">
<script src="js/podo.js" type="text/javascript"></script>
<script src="js/AES.js" type="text/javascript"></script>
<script type="text/javascript" src="stepcounter.js"></script>
<script src="js/notify.js"></script>
<script type="text/javascript" src="cordova.js"></script>
```

Výpis 25: Import JS a CSS v HTML

6.2.2 Pohyb v aplikaci

Aplikace běží v jednom HTML kódu, takže přepínání stránek je vlastně pouze iluze pro uživatele, jelikož se pouze skryje jedná část kódu a ukáže jiná. Tato funkčnost je zajištěna stylováním dokumentu pomocí vlastností „display“ nebo metodami „show“ a „hide“, kde jednoduše zobrazujeme pouze část, kterou uživatel prohlíží.

```
//HTML
<a href="#" id="downloadButton" style="display:none;">Refresh</a>
//JS
$('#hrefHome').on('click',function(){
   $('#downloadButton').hide();
   $('#scanButton').hide();
   $('#historyButton').show();
   $('#mapButton').hide();
   $('#gpxButton').show();
   $('.pages').hide();
   $('#HomePage').show();});
```

Výpis 26: Pohyb v aplikaci

6.2.3 Rozdělení dokumentu na stránky

Dokument je rozdělen do sekcí pomocí párové značky „ “. Tyto sekce v aplikaci slouží prakticky jako stránky, protože stačí vždy zobrazit jednu sekci a ostatní skrýt, a tak vytvořit efekt přepínání stránek. Aplikace obsahuje 5 sekcí a to: Dropdown, HomePage, DevicesPage, BTDevicePage a MapPage. Sekce jsou jednoznačně identifikovatelné pomocí atributu „id“. Tento atribut by měl být vždy jedinečný, hlídá si to však sám vývojář. Pomocí tohoto atributu totiž můžeme dále pracovat s konkrétním elementem třeba v JS kódu nebo element stylovat pomocí CSS. Můžeme také použít element „class“, který se používá pro skupinu elementu, které by měly mít třeba stejnou funkčnost nebo stylování.

```
<div id="MapPage" class="pages" style="display:none;">
   <div id="map"></div>
</div>
```

Výpis 27: Rozdělení HTML dokumentu

Popis jednotlivých sekci:

- Dropdown – Jediná sekce, která nepředstavuje stránku. Tato sekce představuje hlavní nabídku, pomocí které je možné se v aplikaci pohybovat nebo ji obsluhovat.
- HomePage – Sekce představuje hlavní stránku aplikace viz kapitola 5.4.2.
- DevicesPage – Tato sekce představuje stránku pro vyhledávání BT zařízení viz kapitola 5.4.4.

- BTDevicePage – Představuje stránku pro vizualizaci dat z připojeného zařízení viz kapitola 5.4.5.
- MapPage – Sekce pro vizualizaci mapy viz kapitola 5.4.3.

6.3 Přispůsobení vizuálního stylu platformě (CSS)

Dokument CSS se zabývá stylováním dokumentů jako HTML. V této práci jsem se stylování příliš nevěnoval, proto tato kapitola bude kratší.

Cordova ze základu přidá něco málo do CSS kódu, takže nějaké základní stylování stránky existuje od začátku. Při práci s CSS lze udělat spoustu věcí, popíšu zde některé z prvků, které jsem použil v mé výsledné aplikaci.

6.3.1 Identifikace

Než začneme stylovat element je třeba jej prvně identifikovat, neboli vědět, co se vůbec styluje. Pro identifikaci se používají tzv. selektory. Těch je obrovské množství, já jsem ale používal jen ty základní jako název samotného elementu, identifikace podle id nebo třídy.

```
.devices{text-align: center;}
```

Výpis 28: Identifikace elementů v CSS

6.3.2 Stylování

Jak už bylo zmíněno je obrovská škála toho, co a jak se dá stylovat. Po tom, co pomocí selektorů vybereme element, který chceme stylovat, zvolíme vlastnost a nastavíme ji parametry, dle toho, jak chceme, aby element vypadal nebo kde byl umístěn.

Základní vlastnosti:

- height, width – nastaví výšku a šířku elementu, můžeme použít různé jednotky např. pixely (px) nebo procenta (%)
- position – specifikuje metodu použitou u elementu pro pozicování, element s absolutním pozicováním je umístěn relativně vzhledem k nejbližšímu polohovanému předchůdci.
- top, bottom, right, left – nastavení umístění elementu vzhledem ke straně, kterou nastavujeme.
- font-size, font-weight – pomocí těchto vlastností se nastavuje velikost a šířka písma.
- color – udává barvu písma v daném elementu.
- background-color – nastavuje barvu pozadí u elementu.

- border, border-radius – elementu můžeme taky vytvořit okraj, používá se třeba u tabulek. V aplikaci je tato vlastnost použita u ohraničeni odkazu aby připomínal tlačítko. Okraj lze také zaoblit.
- display – již zmíněná vlastnost reprezentující zobrazení elementu.
- overflow – vlastnost určuje, co by se mělo stát, pokud obsah přeplní pole prvku.
- box-shadow – přidává stín k elementu, tato možnost byla přidána s CSS3.
- Z-index – určuje umístění elementu na ose Z, takže zda má být element v popředí či v pozadí.
- padding, margin – nastavují odsazení elementu padding zevnitř a margin zvenku elementu.

```
.dropdown-content {
   display: none;
   position: absolute;
   background-color: #f1f1f1;
   overflow: auto;
   box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
   z-index: 1;}
```

Výpis 29: Stylování (Menu)

6.3.3 Animace

Pomocí CSS můžeme také vytvářet animace. Pravidlo „@keyframes“ specifikuje kód pro animace. Animace je vytvořena postupnou změnou z jedné sady stylů CSS na jinou. Je tedy potřeba vytvořit „@keyframes“ pravidlo, ve kterém specifikujeme začátek a konec animace a poté vytvořením vlastností „animation“ definujeme animaci na elementu. V aplikaci jsem animace použil u vytváření tzv. loaderu, čili prvku který uživateli oznamuje, že aplikace zrovna pracuje.

```
.loader {
   border: 10px solid \#f3f3f3;
   border-radius: 50\%;
   border-top: 10px solid \#3498db;
   animation: spin 2s linear infinite;}
@keyframes spin {
   0\% { transform: rotate(0deg); }
   100\% { transform: rotate(360deg); }}
```

Výpis 30: Animace v CSS (Loader)

7 Testování

Výsledná aplikace byla zveřejněná na GitHub a je dostupná na adrese: https://github.com/rockieDevelop/Stepcounter.

Během testování jsem se zaměřil hlavně na funkčnost na různých zařízeních a spotřebu baterie. Testování probíhalo na třech různých zařízeních na platformě Android. Prvním testovacím zařízením byl mobilní telefon Xiaomi Redmi 3 s Androidem verze 5.1.1 Lollipop, druhým zařízením byl mobilní telefon Samsung Galaxy A5 (2017) s verzi systému Android 8.0.0 Oreo a posledním zařízením, na kterém byla aplikace testována, byl tablet Lenovo TB3-710F s verzi systému 5.0.1 Lollipop.

7.1 Spotřeba baterie

Jelikož aplikace běží na pozadí a celou dobu počítá kroky, neboli čerpá data ať už z vestavěného krokoměru nebo si kroky sama počítá pomocí DeviceMotionEventu a zároveň zaznamenává GPS pozici, tak má značný vliv na spotřebu baterie zařízení.

Dalším prvkem aplikace, který spotřebuje značné množství kapacity baterie je BT komunikace. Během vyhledávání BT zařízení a následného stahovaní dat ze zařízení spotřeba baterie stoupne, proto by tyto operace, pokud možno, měly trvat co nejkratší dobu.

Na zařízení Samsung Galaxy A5 se spotřeba vyšplhala i na 5% baterie za hodinu, u ostatních zařízení byla spotřeba také výrazně větší.

7.2 Bluetooth komunikace

Bluetooth komunikace probíhala na všech zařízeních podle očekávání. Někdy se při této komunikaci stane, že se BT zařízení k mobilnímu zařízení vůbec nepřipojí a komunikace tak vůbec neproběhne, jedná se však o problém technologie, protože to samé se stává i u ostatních aplikací včetně oficiálních aplikací vydaných přímo k těmto zařízením.

Vyhledávání okolních BT zařízení (skenování) nenajde všechny najednou, záleží jak silný signál vydávají (jak dobře je lze vidět), k některým se také nedá připojit a to bohužel zdržuje funkci připojování, protože čeká na vypršení časového limitu pro připojení (tzv. timeout). To jsou důvody, proč vyhledávání vhodného zařízení může trvat delší dobu.

7.3 Rozdíly funkčnosti aplikace u testovaných zařízení

Jak již bylo zmíněno testovacími zařízeními byly mobilní telefon Xiaomi Redmi 3 (dále jen jako Redmi 3), mobilní telefon Samsung Galaxy A5 (dále jen jako A5) a tablet Lenovo TB3-710F (dále jen jako tablet).

Prvním testovaným bylo Redmi 3, zařízení neobsahuje vestavěný krokoměr, proto se kroky počítaly pomocí importované knihovny a DeviceMotionEventu. Tato metoda není tak přesná

Tabulka 2: Porovnání aplikací

Název aplikace	počítání kroků	fitness funkce	koučing	hlídání diety	mapa	synchronizace s applikacemi	komunikace s náramkem
Stepcounter	ano	ano/ne [1]	ne	ne	ano	částečně [2]	ano
Google Fit	ano	ano	ne	ne	ano	ano	ne
Leap Fitness	ano	ano	ne	ne	ne	ne	ne
MyFitnessPal	ano	ano	ne	ano	ne	ne	ne
Pedometer	ano	ano	ne	ne	ne	ne	ne
Pacer Health	ano	ano	ano	ne	ano	ano	ne
Runkeeper	ano	ano	ano	ne	ano	ne	ne
Runtastic Steps	ano	ano	ano	ne	ne	ano	ano

jako čtení kroku z vestavěného krokoměru a neměla by být ani tak robustní (lze ji lehce zmást), ale aplikace zase s touto metodou dokáže počítat průměrnou rychlost a spálené kalorie. Aplikace na zařízení Redmi 3 nedokázala sledovat pozici uživatele, když nebyla připojena k internetu, podle dokumentace technologie pro sledování pozice je v aplikaci napsáno vše správně, což dosvědčuje i to, že na dalších zařízeních lokace fungovala správně. Předpokládám proto, že se jedná o možné omezení přístupu k GPS u systému MIUI (operační systém na bázi Androidu od Xiaomi).

A5 obsahuje vestavěný krokoměr, aplikace ho tedy používá a počítání kroků je tak přesnější a mělo by být i robustnější, neumí však počítat průměrnou rychlost ani spálené kalorie. Při snaze o zmatení přístroje (různé pohyby mobilem do všech stran), se vestavěný krokoměr ukázal robustnější než počítání kroků pomocí importované knihovny, ale ne zcela odolný proti všem pohybům.

Většina uživatelů tabletů používá zobrazení landscape (na šířku), aplikace má však zobrazení uzamčené na portrait (na výšku), aplikace je tedy vhodnější pro uživatelé telefonu, navíc tablety obecně, včetně tabletu, na kterém byla aplikace testována, se nevejdou do kapsy, ale jsou uschovány například v batohu a počítání kroků je potom nepřesné.

7.4 Porovnání s existujícími aplikacemi

Aplikace byla porovnána s aplikacemi z kapitoly 4. Porovnání těchto aplikací s aplikací Stepcounter vytvořenou v rámci této práce lze vidět v tabulce č.2. V tabulce jsou zaznamenány také funkce z plných verzí aplikací, tzn. včetně placených verzí.

[1] U metody s použitím vestavěného krokoměru neobsahuje výpočet rychlosti ani spálených kalorií
[2] Dokáže generovat GPX soubor, který lze dále použít v jiných aplikacích

8 Závěr

Stanovených cílů bylo dosaženo. Výsledná aplikace byla otestována třemi různými zařízeními na platformě Android, umožňuje jak získávání dat z externího senzoru, tak i z pohybových senzorů vestavěných v zařízení. Taktéž umožňuje sledování pozice na mapě a generování GPX souboru.

Aplikace fungovala bez větších nedostatků na všech testovaných zařízeních, dá se tedy předpokládat, že bude fungovat na většině zařízení na platformě Android verze 4.4 a vyšší (ty verze, které podporují BLE komunikaci), která obsahují pohybové senzory pro měření kroků.

Není vhodné nechávat aplikaci běžet neustále i když jí zrovna nepoužíváme, jako třeba v noci, sledování pozice totiž značně snižuje výdrž baterie mobilního zařízení. Stejně tak je náročné na spotřebu vyhledávání okolních BT zařízení, proto by uživatel neměl zapomenout toto vyhledávání zrušit hned po tom co našel hledané zařízení. Snížit spotřebu baterie by bylo možné například snížením frekvence snímání pozic u lokalizace.

Lokalizace pomocí GPS v aplikaci byla přesnější než jsem předpokládal, u zařízení Xiaomi Redmi 3 nefungovalo, nejspíš z důvodu chyby technologie nebo omezení přístupu k GPS systémem MIUI. Při hledání řešení této disfunkce jsem narazil na několik příspěvku jiných vývojářů, řešení však podle mnou nalezených informací zatím není.

Aplikace umí komunikovat pouze s externím senzorem Xiaomi Mi Band 2, po lehké úpravě kódu by však tuto funkčnost mělo být možné přenést i na jiné senzory, aplikace byla testována i na Xiaomi Mi Band 3, což je nástupce výše zmíněného náramku. U tohoto senzoru však nedojde k autentizaci, protože zde byl změněn autentizační protokol a tak se přečtou pouze data o baterii. V kódu by tedy bylo potřeba změnit funkci pro autentizaci náramku. Na rozdíl od existujících aplikací, Stepcounter zaznamenává více druhů aktivit přečtených z externího senzoru jako chůze, běh, spánek, nabíjení zařízení nebo sezení či stání.

Aplikace Stepcounter splňuje stanovené cíle a malé nedostatky, které obsahuje, se v budoucnu dají opravit, jelikož se většinou jedná o chybu technologie, kterou z největší pravděpodobností nové verze opraví a nebo se jedná o potřebu rozšíření zdrojového kódu o doplňující funkce.

Literatura

[1] Martin Chroust, Filip Kůžel *mobilmania.cz* [online], [cit. 21.3.2019], Dostupné z: https://www.mobilmania.cz/clanky/smartphony-maji-19-smyslu-znate-je-vsechny/sc-3-a-1329584/default.aspx

[2] Ryan Kraudel *Quora.com* [online], [cit. 21.3.2019], Dostupné z: https://www.quora.com/How-do-optical-heart-rate-sensors-work

[3] Abhishek Gupta, Imran Mohammed *EDN.com* [online], [cit. 21.3.2019], Dostupné z: https://www.edn.com/5G/4442859/The-basics-of-Bluetooth-Low-Energy–BLE–

[4] Saurabh Pant *AndroidPub* [online], [cit. 21.3.2019], Dostupné z: https://android.jlelse.eu/android-bluetooth-low-energy-communication-simplified-d4fc67d3d26e

[5] Shahar Avigezer *medium.com* [online], [cit. 21.3.2019], Dostupné z: https://medium.com/@avigezerit/bluetooth-low-energy-on-android-22bc7310387a

[6] developer.android.com *developer.android.com* [online], [cit. 21.3.2019], Dostupné z: https://developer.android.com/guide/topics/connectivity/bluetooth-le

[7] developer.android.com *developer.android.com* [online], [cit. 21.3.2019], Dostupné z: https://developer.android.com/about/dashboards

[8] Xiaomi Mi Band 2 *xiaomimarket.cz* [online], [cit. 21.3.2019], Dostupné z: https://www.xiaomimarket.cz/xiaomi-mi-band-2/

[9] pcmag.com *pcmag.com* [online], [cit. 21.3.2019], Dostupné z: https://www.pcmag.com/encyclopedia/term/63822/android-versions/

[10] Forrest Stroud *webopedia.com* [online], [cit. 21.3.2019], Dostupné z: https://www.webopedia.com/TERM/A/android_codenames.html

[11] Mašek, Pavel *Vývoj multiplatformních mobilních aplikací*, Hradec Králové, 2017 [cit. 2019-03-21]. Dostupné z: https://theses.cz/id/20k2p8. Diplomová práce. Univerzita Hradec Králové. Fakulta informatiky a managementu. Katedra informatiky a kvantitativních metod. Vedoucí práce Ing. Jiří Štěpánek.

[12] Ulip, Marek *Aplikace pro záznam a vizualizaci dat senzorů na platformě Android*, Ostrava, 2018 [cit. 2019-03-21]. Dostupné z: http://hdl.handle.net/10084/128699. Bakalářská práce. Vysoká škola báňská - Technická univerzita Ostrava.

[13] Krosapp *krosapp.cz* [online], [cit. 21.3.2019], Dostupné z: http://www.krosapp.cz/Blog/multiplatformni-vyvoj-mobilnich-aplikaci-druhy-nastroje/

[14] Techopedia *techopedia.com* [online], [cit. 21.3.2019], Dostupné z: https://www.techopedia.com/definition/3929/javascript-js

[15] developer.mozilla.org *developer.mozilla.org* [online], [cit. 21.3.2019], Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTML

[16] developer.mozilla.org *developer.mozilla.org* [online], [cit. 21.3.2019], Dostupné z: https://developer.mozilla.org/en-US/docs/Web/CSS

[17] Apache Cordova *cordova.apache.org* [online], [cit. 21.3.2019], Dostupné z: https://cordova.apache.org/docs/en/latest/guide/overview/index.html

[18] Vangie Beal *webopedia.com* [online], [cit. 21.3.2019], Dostupné z: https://www.webopedia.com/TERM/A/API.html

[19] Wikipedia *en.wikipedia.org* [online], [cit. 21.3.2019], Dostupné z: https://en.wikipedia.org/wiki/Plug-in_(computing)

[20] Apache Cordova *cordova.apache.org* [online], [cit. 21.3.2019], Dostupné z: https://cordova.apache.org/plugins/

[21] gpsnavigace.cz *gpsnavigace.cz* [online], [cit. 21.3.2019], Dostupné z: http://www.gpsnavigace.cz/prispevky/co_je_gps.htm

[22] topografix.com *topografix.com* [online], [cit. 21.3.2019], Dostupné z: https://www.topografix.com/gpx.asp

[23] developer.mozilla.org *developer.mozilla.org* [online], [cit. 21.3.2019], Dostupné z: https://developer.mozilla.org/en-US/docs/Web/XML/XML_introduction

[24] W3C *w3.org* [online], [cit. 21.3.2019], Dostupné z: https://www.w3.org/XML/

[25] Sébastien Ménigot *sebastien.menigot.free.fr* [online], [cit. 25.3.2019], Dostupné z: http://sebastien.menigot.free.fr/pedometer_explanations.html

[26] Antonín Vojáček *automatizace.hw.cz* [online], [cit. 25.3.2019], Dostupné z: https://automatizace.hw.cz/clanek/2007042901

A Přílohy v ZIP souboru

Přiložený ZIP soubor obsahuje:

1. Zdrojové kódy aplikace
2. Výsledné APK aplikace Stepcounter